Abyan Ardiatama

24060120140161

Praktikum 5 ML

Principal Component Analysis

Import Dataset

--> Blood Transfusion

```
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```

```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/blood-transfus
names = ['Recency', 'Frequency', 'Monetary', 'Time', 'Class']
dataset = pd.read_csv(url,names=names)
dataset
```

	Recency	Frequency	Monetary	Time	Class
0	Recency (months)	Frequency (times)	Monetary (c.c. blood)	Time (months)	whether he/she donated blood in March 2007
1	2	50	12500	98	1
2	0	13	3250	28	1
3	1	16	4000	35	1
4	2	20	5000	45	1
...	...	...	...	...	...
744	23	2	500	38	0
745	21	2	500	52	0
746	23	3	750	62	0
747	39	1	250	39	0
748	72	1	250	72	0

749 rows × 5 columns

```
#cleaning data
dataset = dataset[dataset.Recency != 0]
dataset = dataset[dataset.Frequency != 0]
dataset = dataset[dataset.Monetary != 0]
dataset = dataset[dataset.Time != 0]
```

Konversi categorical value menjadi numerical value dalam dataset

```
#convert all categorical features into numerical values
dataset['Recency'] = pd.to_numeric(dataset['Recency'], errors='coerce')
dataset['Frequency'] = pd.to_numeric(dataset['Frequency'], errors='coerce')
dataset['Monetary'] = pd.to_numeric(dataset['Monetary'], errors='coerce')
dataset['Time'] = pd.to_numeric(dataset['Time'], errors='coerce')
dataset['Class'] = pd.to_numeric(dataset['Class'], errors='coerce')
print(dataset.dtypes)
```

```
    Recency      float64
    Frequency    float64
    Monetary     float64
    Time         float64
    Class        float64
    dtype: object
```

Standarisasi fitur dalam dataset

```
features = ['Recency', 'Frequency', 'Monetary', 'Time']
# Separating out the features
x = dataset.loc[:, features].values
# Separating out the target
y = dataset.loc[:,['Class']].values
# Standardizing the features
x = StandardScaler().fit_transform(x)
x = x[~np.isnan(x).any(axis=1)]
x
```

```
    array([[-0.93703771,  7.67939033,  7.67939033,  2.61934918],
           [-1.06080998,  1.81438569,  1.81438569,  0.02948995],
           [-0.93703771,  2.50438623,  2.50438623,  0.44057872],
           ...,
           [ 1.66217996, -0.42811609, -0.42811609,  1.13942962],
           [ 3.64253629, -0.77311636, -0.77311636,  0.19392545],
           [ 7.7270212 , -0.77311636, -0.77311636,  1.55051839]])
```

Menghapus semua missing value, mengecek adanya infinity value dalam dataset

```
#remove all rows with missing values
dataset = dataset.dropna()
dataset.shape
dataset.isnull().sum()
```

```
Recency      0
Frequency    0
Monetary     0
Time         0
Class        0
dtype: int64
```

```
import pandas as pd
import numpy as np

# checking for infinite values and displaying the count
count = np.isinf(dataset).values.sum()
print("Infinity values... ",count)
```

```
Infinity values...  0
```

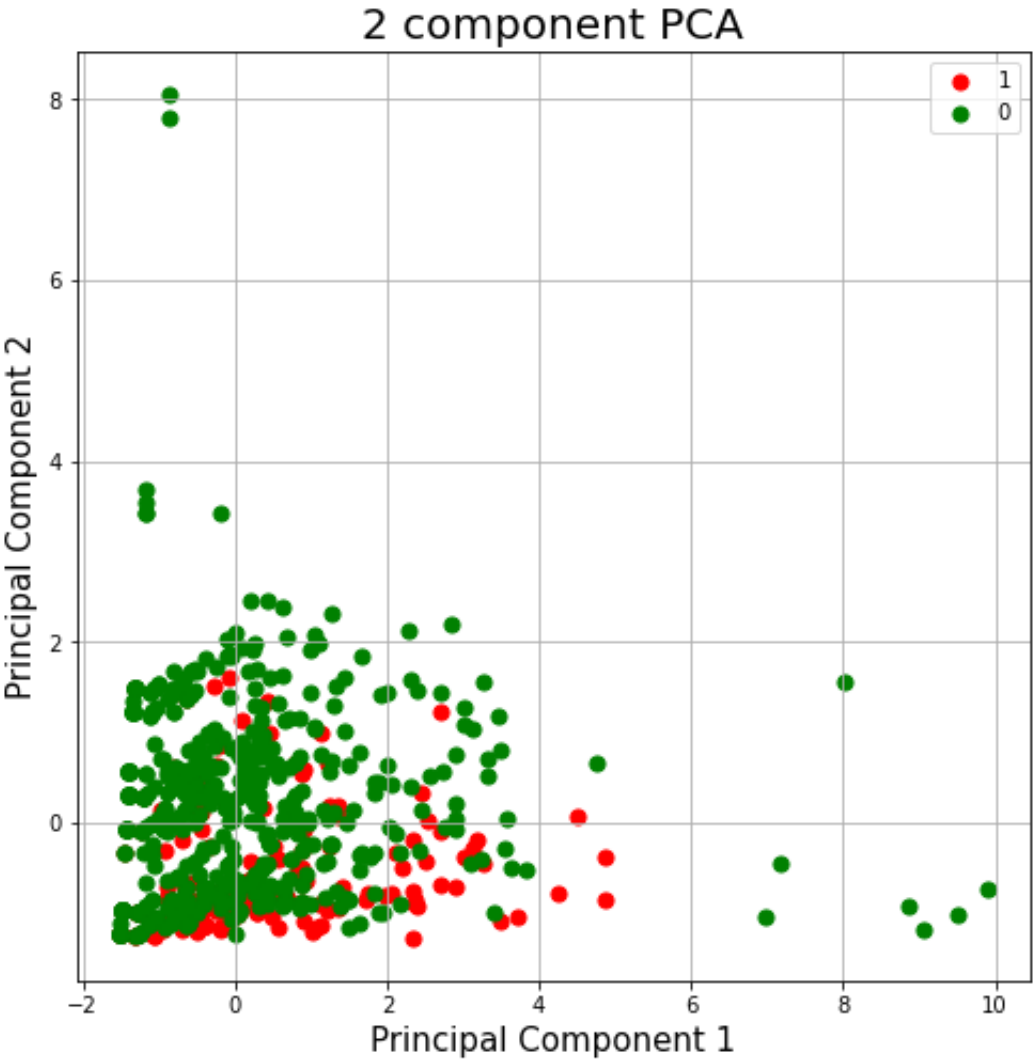
Memproyeksikan PCA ke dalam 2 Dimensi

```
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents, columns = ['principal com
```

```
finalDf = pd.concat([principalDf, dataset[['Class']]], axis = 1)
```

Visualisasi

```
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2 component PCA', fontsize = 20)
targets = [1, 0]
colors = ['r', 'g']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['Class'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
ax.legend(targets)
ax.grid()
```



```
pca.explained_variance_ratio_
```

```
    array([0.6348139 , 0.27530624])
```

Colab paid products - Cancel contracts here

✓ 0s completed at 16:29 ● ✕